TransactIQ Team Iteration One

Meet the Team

- Chandler Clevenger
- Robert Duncan
- Jake Freestone
- Jay Ling
- Robert Mooney

Client Company



Mentor Feedback

- Would like to see better error handling
- Need to use SRP for one of the files for it has more than one class
- Better styling and gave us resources
- Gave us mapping libraries to use for back-end

TransactIQ Team Iteration One

Meet the Team

- Chandler Clevenger
- Robert Duncan
- Jake Freestone
- Jay Ling
- Robert Mooney

Client Company

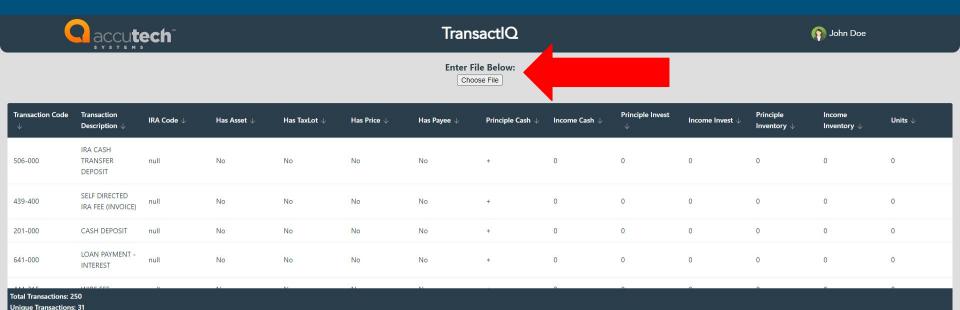


Mentor Feedback

- Would like to see better error handling
- Need to use SRP for one of the files for it has more than one class
- Better styling and gave us resources
- Gave us mapping libraries to use for back-end

Client Feedback

- Really liked it
 - Visually looked good
 - Favored the sorting functionality
 - Surpassed expectations for Iteration One
- One of our usual clients wasn't there



Functional Requirement 1

The system shall take in a file or user input.

Priority level: HIGH



TransactIQ



Enter File Below:

Choose File

Transaction Code ↓	Transaction Description ↓	IRA Code ↓	Has Asset ↓	Has TaxLot ↓	Has Price \downarrow	Has Payee ↓	Principle Cash ↓	Income Cash \downarrow	Principle Invest	Income Invest ψ	Principle Inventory ψ	Income Inventory ↓	Units ↓
506-000	IRA CASH TRANSFER DEPOSIT	null	No	No	No	No	***	0	0	0	0	0	0
439-400	SELF DIRECTED IRA FEE (INVOICE)	null	No	No	No	No	+	0	0	0	0	0	0
201-000	CASH DEPOSIT	null	No	No	No	No	+	0	0	0	0	0	0
641-000	LOAN PAYMENT - INTEREST	null	No	No	No	No	+	0	0	0	0	0	0
0.0000 900000													102

Total Transactions: 250 Unique Transactions: 31

Functional Requirement 2

The system shall display the characteristics it finds within the data.

Priority level: HIGH

Functional Requirement 3

Priority level: MEDIUM

The system shall be a web application.



https://localhost:5002

vue3 project



Backend

localhost:7225

dotnet core web api



TransactIQ



Enter File Below:

Choose File

Transaction Code ↓	Transaction Description ↓	IRA Code \downarrow	Has Asset ↓	Has TaxLot ↓	Has Price ↓	Has Payee ↓	Principle Cash ↓	Income Cash ↓	Principle Invest	Income Invest ↓	Principle Inventory ↓	Income Inventory ψ	Units ↓
506-000	IRA CASH TRANSFER DEPOSIT	null	No	No	No	No	+	0	0	0	0	0	0
439-400	SELF DIRECTED IRA FEE (INVOICE)	null	No	No	No	No	+	0	0	0	0	0	0
201-000	CASH DEPOSIT	null	No	No	No	No	+	0	0	0	0	0	0
641-000	LOAN PAYMENT - INTEREST	null	No	No	No	No	u+	0	0	0	0	0	0
444.245	WHO E EEE		A.L.					-0	0				

Total Transactions: 250
Unique Transactions: 31

Non-functional Requirement 1

The system shall have a distinct look in the UI.

Priority level: LOW

Non-functional Requirement 2

Priority level: MEDIUM

The UI and Logic must be separate.

Deploys with: npm run serve

Frontend

https://localhost:5002

vue3 project

Deploys with: dotnet --project ./back-end/

Backend



localhost:7225

dotnet core web api

▷ ✓ 🖨 back-end
▷ △ 🚍 back-end.Tests
▷ △ 🖶 front-end

Planned Iteration Two Features

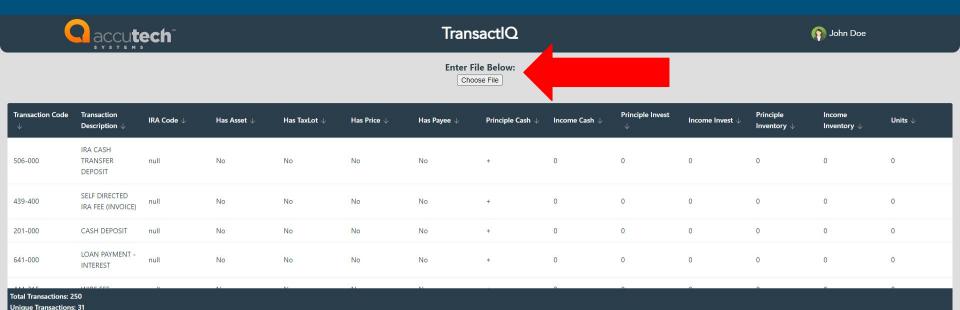
- Docker Integration
- Database connection
- Login Page
- User creation, management, and authentication
- Basic data analysis



Demonstration

Client Feedback

- Really liked it
 - Visually looked good
 - Favored the sorting functionality
 - Surpassed expectations for Iteration One
- One of our usual clients wasn't there



Functional Requirement 1

The system shall take in a file or user input.

Priority level: HIGH



TransactIQ



Enter File Below:

Choose File

Transaction Code ↓	Transaction $egin{array}{c} Description \ \downarrow \end{array}$	IRA Code ↓	Has Asset ↓	Has TaxLot ↓	Has Price ↓	Has Payee ↓	Principle Cash ↓	Income Cash $ \downarrow $	Principle Invest	Income Invest ψ	Principle Inventory ↓	Income Inventory ↓	Units ↓
506-000	IRA CASH TRANSFER DEPOSIT	null	No	No	No	No	**	0	0	0	0	0	0
439-400	SELF DIRECTED IRA FEE (INVOICE)	null	No	No	No	No	+	0	0	0	0	0	0
201-000	CASH DEPOSIT	null	No	No	No	No	+	0	0	0	0	0	0
641-000	LOAN PAYMENT - INTEREST	null	No	No	No	No	+	0	0	Ö	0	0	0
0.000 - 0.000													20

Total Transactions: 250 Unique Transactions: 31

Functional Requirement 2

The system shall display the characteristics it finds within the data.

Priority level: HIGH

Functional Requirement 3

Priority level: MEDIUM

The system shall be a web application.



https://localhost:5002

vue3 project



Backend

localhost:7225

dotnet core web api



TransactIQ



Enter File Below:

Choose File

Transaction Code ↓	Transaction Description ↓	IRA Code \downarrow	Has Asset ↓	Has TaxLot ↓	Has Price ↓	Has Payee ↓	Principle Cash ↓	Income Cash ↓	Principle Invest	Income Invest ↓	Principle Inventory ↓	Income Inventory ψ	Units ↓
506-000	IRA CASH TRANSFER DEPOSIT	null	No	No	No	No	+	0	0	0	0	0	0
439-400	SELF DIRECTED IRA FEE (INVOICE)	null	No	No	No	No	+	0	0	0	0	0	0
201-000	CASH DEPOSIT	null	No	No	No	No	+	0	0	0	0	0	0
641-000	LOAN PAYMENT - INTEREST	null	No	No	No	No	u+	0	0	0	0	0	0
444.245	WHO E EEE		A.L.						0				

Total Transactions: 250
Unique Transactions: 31

Non-functional Requirement 1

The system shall have a distinct look in the UI.

Priority level: LOW

Non-functional Requirement 2

Priority level: MEDIUM

The UI and Logic must be separate.

Deploys with: npm run serve

Frontend

https://localhost:5002

vue3 project

Deploys with: dotnet --project ./back-end/

Backend



localhost:7225

dotnet core web api

▷ ✓ 🖨 back-end
▷ △ 🚍 back-end.Tests
▷ △ 🖶 front-end

Planned Iteration Two Features

- Docker Integration
- Database connection
- Login Page
- User creation, management, and authentication
- Basic data analysis



Demonstration